Teaching Statement

Joshua Reich

As a teacher, I find great satisfaction in helping students master new material and skills, both inside and outside of the classroom. My goal is to encourage students to grow as critical thinkers and computer science professionals. I focus on conveying concepts and developing each student's ability to ask the right questions.

In the Classroom

During my PhD, I taught Columbia University's COMS W3101-2: Programming Languages C++[1], both on campus (two semesters) and online (five semesters). I designed this course from scratch: developing interactive programming tutorials, lecture notes, programming assignments, exams, and grading rubrics. I delivered all lectures, held regular office hours, and managed teaching assistants.

COMS W3101-2 is an introductory-level class attended by a wide cross-section of students. My classes comprised undergraduates, graduates, and continuing education students. Their disciplines ranged from computer science to business to biochemistry. For many, this was their first experience programming, while others had coded for decades.

To address this diverse population, my lectures interactively mixed slide-based presentation and live tutorial. I would illustrate each new concept, or language construct, by editing, compiling, and executing C++ programs. All example programs were posted online, and I welcomed students to code along with me. I would intentionally introduce bugs into the code—soliciting corrections from the class and demonstrating the use of debugging tools such as `gdb`, when programs failed to compile or execute as expected. By analyzing how program execution differed from their expectations, students learned hands-on that programs execute according to specifications, not intentions.

Grounding my lecture slides in live examples allowed me to keep the lectures fast-paced, while providing less experienced or slower-learning students the support they required. Moreover, in addition to frequently stopping for questions, lecture videos were posted online, allowing each student to revisit the lectures at her own pace.

Homework problems required students to generate clearly commented, runnable code. Each problem was broken into several stages, with the credit for each stage clearly marked. Later homework sets featured an optional challenge problem. This structure enabled less advanced students to feel a sense of progress and accomplishment, while still challenging advanced students.

The final exam's format matched those of the homework. During the exam, students had full access to their notes, all course materials (including lecture videos), their preferred IDEs and debugging tools, and the web. My goal was to produce students who could figure out the required program logic, utilize appropriate C++ features, and produce the necessary syntax to encode this logic—not ones who had memorized every piece of the C++ syntax or library API that they might conceivably need.

Providing actionable feedback is a critical part of teaching and I designed lectures, homework, and the final exam with this in mind. However, it was my responsibility to assess students for certification as well. Grading student performance accurately and fairly is a serious responsibility. While students had unrestricted internet access for the final exam, they were prohibited from using messaging clients, email, or other communication mechanisms during the exam. Exams were conducted in a computing lab where both I and course TAs could monitor for the presence of prohibited communication. I utilized automated tools like Moss[2] on both homework and exams to detect potential violation of the course cheating policy.

Additionally, I have served as a teaching assis-

[1] `http://www.cs.princeton.edu/~jreich/cs3101`

[2] `http://theory.stanford.edu/~aiken/moss/`

tant for three separate courses. As an undergraduate, I TA'd Columbia's introduction to programming. This role was relatively limited, and focused on grading. Thus I was excited when the opportunity arose to take a more hands-on role as head TA for two graduate level courses, Columbia University's COMS W4205: Combinatorial Theory and COMS W4203: Graph Theory. In addition to grading, I created both course websites, ran tutorial sessions, filled in as lecturer, managed other TAs, and held office hours. Students recognized my efforts; based on their rankings, the School of Engineering and Applied Science selected me for back-to-back Extraordinary Teaching Assistant Awards.

As a postdoc, I have assisted in developing new courses. When Nick Feamster contacted me to express interest in using my Pyretic SDN programming platform in his Coursera SDN Massive Open Online Course[3] (enrollment ∼50K), I volunteered to support those portions that involved Pyretic. I designed the Pyretic assignment module (completed by ∼1K students), provided lecture material, and monitored the course discussion board.

The following semester, I helped shape Princeton COS 597E: Software Defined Networking, taught by my mentor, Jen Rexford. Many of the student projects in COS 597E utilized Pyretic and I met with project groups frequently to provide guidance and technical support. I have also contributed to similar courses at other universities.

I have a broad background in Computer Science and am comfortable teaching a wide range of introductory classes. I am particularly qualified to teach advanced classes in systems and networking, and seminars in software defined networking and cloud computing. If needed, I could also teach advanced courses in graph theory, and combinatorial theory. I hope to develop a new project-centric course combining software design, prototyping, intellectual property, and entrepreneurship.

Outside of the Classroom

Some of the most exciting teaching happens outside of the classroom—in one-on-one discussions, brainstorming sessions, reading groups, and hackathons. As a faculty member, I want to foster an environment rich with opportunities for such activity. At Columbia, I ran the joint CS/EE networking seminar: inviting speakers, setting up meeting schedules, and advertising talks. I helped organize student-run reading groups in network-coding (at Columbia) and software-defined networking (at Princeton). I also organize frequent multi-hour Pyretic hack sessions at Princeton, giving students a friendly space to get my assistance, learn from one another, eat free food, break out into impromptu discussions, and, of course, code.

In my view, the PhD is fundamentally closer to a classical apprenticeship than a modern Bachelors or Masters program, which at most institutions are heavily course-centric. Even for non-PhD students, a mentoring relationship in the context of a research course or thesis can be one of the most influential experiences of their degree. I deeply enjoy advising and mentoring self-motivated students. I have supervised, mentored, or advised close to 20 undergraduate, masters, and PhD students during the past half-decade. My students' work has resulted in academic publications, including an award-winning SIGCOMM poster. I have helped students navigate the PhD application process, supported them in their industry job searches, and encouraged them to explore horizon-expanding opportunities such as internships. Moreover, I have focused on doing so in ways that are sensitive to their quality of life.

The educational experience at a top university can be an emotionally challenging one, especially for international students who are far from home. I believe strongly that faculty and departments should make greater efforts to provide students with opportunities to learn soft skills. Examples include stress management, recognizing and dealing with depression, business etiquette training, time management and overcoming procrastination. As a faculty member at your school, I would be excited to help develop programs and support groups for students addressing these issues.

Conclusion: Ultimately, the same philosophy drives both my approach to research and education. I *explore through implementation*, developing innovative strategies for teaching and learning (e.g., Delay Tolerant Talk Prep[4]). I *emphasize practicality in design*, focusing on the needs of my students inside and outside of the classroom.

[3] `https://www.coursera.org/course/sdn`

[4] `http://www.layer9.org/2013/04/exploiting-your-social-network-through.html`